IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal

Received 4 September 2023, accepted 23 October 2023, date of publication 27 October 2023, date of current version 3 November 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3328246

RESEARCH ARTICLE

Multi-Path Minimum Spanning Tree and Superpixel Based Cost Aggregation for Stereo Matching

LONGHAO SUN
College of Computer Science and Technology, Harbin Engineering University, Harbin 150001, China
e-mail: sunlonghao2000@163.com

ABSTRACT Cost aggregation is a key step in stereo matching algorithms. Despite more than a decade of development, most algorithms still encounter challenges such as high error rates in low-texture regions and blurred edges. To improve matching accuracy, we propose a novel cost aggregation method based on multi-path minimum spanning tree (mPMST) and superpixel in this paper. The mPMST offers more optional paths for cost aggregation than the original MST by treating the reference image as an eight-connected graph. To improve both accuracy and computational efficiency, we innovatively run the mPMST for cost aggregation at the inside-superpixel level and superpixel level, which can obtain high accuracy in high-texture regions and low-texture regions respectively. In order to effectively fuse the two-level aggregated costs, we propose a novel adaptive weight based on calculating image entropy for each superpixel. This method can distinguish between regions with high and low texture and quantify texture complexity. Additionally, a novel disparity map refinement method is proposed to improve the quality of disparity maps using the novel cost aggregation structure proposed. In the experimental studies, we test our method on Middlebury and KITTI benchmarks. Average error rates of 5.94% for Middlebury 2006 and 24.51% for KITTI 2015 are achieved. Our experiments show improvement in accuracy compared with other state-of-the-art approaches.

INDEX TERMS Stereo matching, minimum spanning tree, disparity map refinement, cost aggregation, simple linear iterative clustering superpixel segmentation.

I. INTRODUCTION

Dense two-frame stereo matching has been one of the most widely studied fundamental problems in computer vision. The input for stereo matching is a pair of images from the same scene acquired by a stereo camera system, and their epipolar lines are be rectified. Image pairs comprise the left and right views. Because their rectified epipolar lines are horizontal, the search of pixel correspondences between the image pair can be performed along horizontal lines as shown in Figure 1 by the two red matching points. The disparity is the difference between the horizontal coordinates of a pair of corresponding pixels. The aim of stereo matching is to compute a disparity map between the input stereo image pairs, and the disparity map can be used to reckon the depth information via the triangulation principle. Solak and Bolat [36] proposed a hybrid stereovision-based distance-estimation approach. They achieve high accuracy in calculating distance in mobile robot platforms. Thanks to excellent research in the past decades, stereo matching has been applied to many high-level computer vision tasks such as mobile robot navigation [1], obstacle detection and autonomous driving [2], pose estimation and recognition [3], target detection and 3D scene reconstruction [4], UAV localization and navigation [5], and other areas [6], [7], [8]. Due to the ambiguity of the matching problem and the presence of noise, occlusion, or low-texture regions in images, accurate stereo matching is very challenging. Consequently, the pursuit of accurate disparity maps remains an ongoing endeavor.

According to the analysis and classification scheme proposed in [9], stereo matching algorithms can be divided into two categories: local algorithms and global algorithms.

The associate editor coordinating the review of this manuscript and approving it for publication was Kumaradevan Punithakumar.

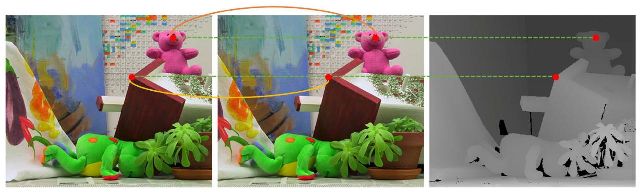
FIGURE 1. The Teddy in Middlebury [38]. From left to right: the left view (the reference image), the right view, the disparity map.

Stereo matching algorithms usually follow a subset of these four steps or all of them to implement:

1. Calculation of the cost volume.
2. Cost aggregation.
3. Disparity computation or optimization.
4. Disparity refinement.

In the calculation of the cost volume phase, an initial 3D cost volume is generated by calculating matching costs for each pixel at all possible disparity levels. In cost aggregation, all matching costs of each pixel are aggregated within a support window. Subsequently, the aggregated cost of each pixel is calculated by global or local optimization methods. In this paper, we employ winner-take-all (WTA) algorithm, selecting the disparity with the lowest aggregated cost for each pixel as the final disparity. Finally, the initial disparity map is refined via various post-processing methods to yield a better disparity map. Among the four steps of stereo matching algorithms, the step of cost aggregation is a pivotal determinant of disparity map quality, particularly within the realms of local and non-local algorithms. The global algorithms usually don't need cost aggregation. Popular global methods are dynamic programming [10], [11], belief propagation [12], [15], [16] and graph cuts [13], [30]. For local algorithms, the disparity calculation at a given image pixel only relies on intensity or color values within a specific window. They usually make implicit smoothing assumptions through aggregation support. On the other hand, global algorithms make explicit smoothing assumptions and then solve the optimization problem. Such algorithms usually omit the cost aggregation step and instead seek a disparity solution that minimizes the global cost function. This paper mainly focuses on cost aggregation.

Cost aggregation entails the local summation or averaging of matching costs for nearby pixels with similar disparity values. An efficient local cost aggregation method is a non-normalized box filter, which runs in linear time (relative to the number of pixels) using integral images [14]. Nonetheless, this method suffers from blurring depth edges. Yoon and Kweon [37] demonstrated that edge-aware filters such as bilateral filter [17] are useful for preserving depth edges, and Yang used the bilateral filter for depth super-resolution. However, the full kernel implementation of the bilateral filter is very slow. Reducing computational complexity, several approximation methods [19], [20], [21] for accelerating the bilateral filter have been proposed. Hosni et al. [22] utilized the guided image filter (GF) [23] for cost aggregation, a linear-time method outperforming most local cost aggregation algorithms in terms of both speed and accuracy. This filter output is a linear transformation of pixels within local support windows. Liu and Li et al. proposed a shape-adaptive bootstrap filter and stereo matching of windows based on texture properties [32], [33], which enhance results in regions with repeating texture. However, the size of local windows used in this algorithm determines the computational complexity as well as the matching accuracy. Now it is still difficult to determine the best window size in practical applications.

In the field of non-local algorithms, Yang [24], [25] proposed a non-local cost aggregation method for the first time by constructing a minimum spanning tree (MST) on a reference image, enabling each pixel to gather support information in the entire image along the tree's shortest path. Unlike previous local methods that rely on local support windows, non-local methods perform cost aggregation for each pixel in the entire image. Considering that the tree structure of MST is not unique because there are many edges with the same weights in an image. By enforcing tight connections for the pixels inside the same segment, Mei et al. [26] proposed a segment tree (ST) structure to perform non-local cost aggregation instead of MST. ST can build a unique tree structure in an image. However, the running speed of ST is very slow. Tung et al. [38] proposed an effective hybrid tree structure that can improve the matching accuracy in low texture regions, but it is still limited by the paths of the tree structure, which makes it difficult to further improve the matching accuracy. Cheng et al. [27] introduced a cross tree structure comprising a horizontal tree and a vertical tree for cost aggregation. Two non-local cost aggregations are accomplished by traversing two cross trees consecutively. However, because [28], [29] are cross tree structures, the connectivity between two neighboring pixels in the diagonal direction cannot be maintained. Subsequently, Chai et al. [40] combined semi-global and MST for stereo matching. Zhang et al. [18] proposed an edge-preserving minimum spanning tree (EMST). This method can preserve edges by changing the weight of edges that cross Canny edges. However, this method has little improvement in accuracy for low-texture regions. Jin et al. [39] proposed a spatial MST filter. This filter increases paths for cost aggregation and uses recursive algorithms to speed up operation. Therefore, it can achieve all pixels in an entire image and provide information for a single pixel. However, not all pixels can provide effective and useful support information. The similar pixels usually provide more effective information to each other. Although this filter can improve accuracy and preserve edges, a pixel often receives a lot of noise and useless information from other pixels. It is difficult to further improve accuracy.

In summary, for all non-local cost aggregation methods, low accuracy in low-texture regions and blurring of depth boundaries are challenging issues.

In this paper, we propose a cost aggregation method based on multi-path minimum spanning tree and superpixel. This method not only maintains high matching accuracy in low-texture regions but also can effectively solve the problems of difficult matching in low-texture regions and blurry object edges. In a reference image, each pixel is regarded as a node in an undirected eight-connected graph. Consequently, it provides more paths to select for cost aggregation compared to Yang's MST. However, constructing an mPMST in the entire image is challenging due to the substantial time investment, and the accuracy improvement in low-texture regions is not significant. We run the mPMST inside each superpixel, which can notably reduce running time and improve accuracy in high-texture regions owing to more optional paths. Through observation of Figure 7, it becomes evident that cost aggregation among superpixels can obtain accurate support information in low-texture regions. Hence, we run the mPMST at superpixel level to improve accuracy in low-texture regions. In order to fuse the two aggregated costs, we propose a novel adaptive weight based on image entropy, serving to quantify texture complexity. Lastly, the proposed novel cost aggregation structure is employed to refine initial disparity maps. The aggregated costs of unstable pixels can be recalculated based on stable pixels, subsequently rectifying the erroneous aggregated costs. The main contributions of this paper are as follows:

1. A novel tree structure mPMST is proposed for cost aggregation. The mPMST can allow each pixel to receive information from more paths because it is built in an eight-connected graph. To minimize the error rate and running time, we creatively execute mPMST at both the inside-superpixel and superpixel levels.

2. Through the novel tree structure, we propose a novel disparity refinement method to further improve the quality of disparity maps. The unstable pixels can be supported by stable pixels along tree paths. The aggregated value of each unstable pixel can be recalculated.

3. In order to assess the texture complexity of each superpixel, a novel adaptive weight fusion by calculating image entropy is proposed. This weight guarantees the effective fusion of two-level aggregated costs.

The remainder of this paper is organized as follows: we reviewed previous related works in Section II. We describe the proposed stereo matching method in Section III. The experimental results and discussions are presented in Section IV. We draw our research conclusions in Section V.

II. RELATED WORKS

Over the past decades, stereo matching has been intensively researched, and a lot of methods have been proposed to improve the quality of disparity maps. In this section, we mainly review non-local cost aggregation methods.

Yang et al. [24], [25] proposed a non-local cost aggregation method by constructing a minimum spanning tree. They depict an image as an undirected four-connected graph as shown in subgraph (a) in Figure 2, where each pixel is treated as a node, the edges connect adjacent pixels in four directions. The weight of each edge is calculated by the color differences between the connected pixels. The similarity between any two pixels is determined by their edge weight. The MST is constructed by iteratively selecting the lowest edge weight in a graph. An edge with low weight is generally unlikely to cross the depth boundary because of the high similarity between its two endpoints. Cost aggregation can be efficiently executed by traversing the tree structure in two steps: from the leaf node to the root node and from the root node to the leaf node. Therefore, each pixel can receive support information from other pixels along the shortest path. The structure of the minimum spanning tree is shown in subgraph (b) of Figure 2. However, this approach is sensitive to noise and there may have many edges with the same weight leading to an unstable tree structure. Furthermore, since the graph is four-connected, it is impossible to select diagonal edges so that some spatial information between neighboring pixels is ignored. This method does not consider the edge features in an image. The prevalent issues for existing MST-based stereo matching methods are that the construction of MST may have inaccurate disparity estimation in low-texture regions and depth discontinuity regions.

Chang et al. [27] proposed a cross tree structure as shown in Figure 2 (c). It can achieve cost aggregation in both horizontal and vertical directions for the central pixel. This method incorporates edge information in calculating edge weights and changed tree structure, yet still tends to blur the edges. This structure is sensitive to noise, and edges can cross depth discontinuity boundaries because the tree structure is fixed, although adding prior information such as edge extraction to prevent blurring in depth discontinuous boundaries.

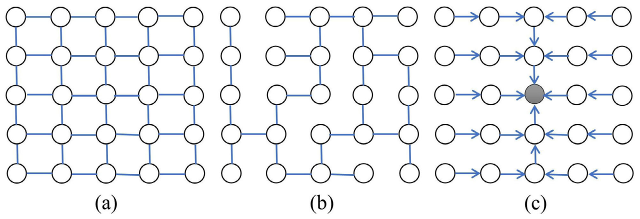

FIGURE 2. Different tree structures of cost aggregation methods. (a) A four-connected graph. (b)The structure of MST. (c)The structure of the cross tree.

Zhang et al. [18] proposed an EMST by using the image edges and brightness information. This method needs to utilize the Canny algorithm to extract edges at first. They designed a self-adaptive function to increase the weights of edges crossing the Canny edges. Therefore, it is more difficult for paths on an EMST to cross depth boundaries than Yang's MST. The EMST addresses the issue of blurry edges to some extent and makes each pixel receive more useful information from other pixels. However, in low-texture regions, the accuracy improvement remains modest and heavily reliant on post-processing algorithms.

Inspired by these methods, we can find that introducing edge information and changing aggregation paths are effective ways to improve accuracy. Therefore, we propose a novel tree structure to address these challenging problems such as edge blurring and low accuracy in low-texture regions.

We involve superpixels to limit the scope of cost aggregation and increase the optional paths to ensure each pixel receives more appropriate support information. Our method can adaptively fuse the two-level aggregated costs to improve accuracy in low-texture regions. Our method differs from related methods in terms of utilizing edge information and processing in regions with low texture.

III. OUR METHOD

Motivated by the advantages and disadvantages of some previous research achievements such as MST, cross tree, and so on. we propose a novel cost aggregation method for stereo matching. This section first describes the proposed multi-path minimum spanning tree, and then describes how it operates inside and among superpixels. Finally, a new disparity refinement method is introduced in this section. The fundamental procedure of our method is shown in Figure 3.

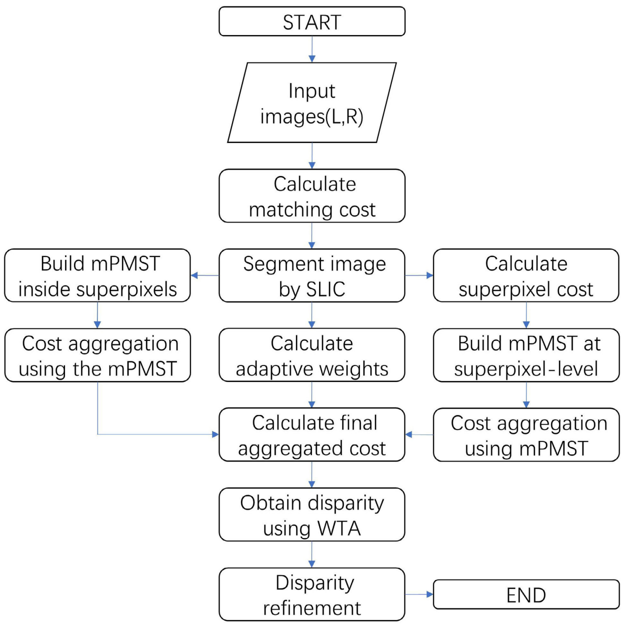
START
Input images(L,R)
Calculate matching cost
Build mPMST inside superpixels
Segment image by SLIC
Calculate superpixel cost
Cost aggregation using the mPMST
Calculate adaptive weights
Build mPMST at superpixel-level
Calculate final aggregated cost
Cost aggregation using mPMST
Obtain disparity using WTA
Disparity refinement
END

FIGURE 3. Flow chart of the proposed stereo matching method.

We calculate the matching cost for each pixel in the image L, and then use SLIC to achieve superpixel segmentation. The edges extracted by SLIC can be well attached to the image object boundaries. The number of superpixels depends on the image size. The SLIC also can segment images with linear time complexity. Subsequently, the novel tree structure proposed for cost aggregation is constructed and executed inside each superpixel. Simultaneously, the superpixel-level tree is also established and executed among all superpixels. Finally, the two aggregated costs of each pixel are fused according to its adaptive weight. Each pixel can obtain an accurate aggregated cost. An initial disparity map is generated using the WTA algorithm. Nevertheless, there are often many black holes like the subgraph (d) of Figure 8 and inaccurate disparity values. We utilize a novel disparity refinement method proposed in this paper to optimize the initial disparity map.

A. MATCHING COST CALCULATION AND MPMST

For cost aggregation, a 3D cost volume is essential, so we need to calculate the matching cost of each pixel at all possible disparity levels to establish a 3D cost volume.

We assume that the left view of a stereo image pair is L, the right view is R, and D is a set containing all optional disparity values, with $|D|$ representing the total number of elements in this set. Generally, the left view serves as a reference image. The matching cost of pixel p in the image L with a certain disparity d can be donated by $C_d(p)$, and the pixel p can be represented by their coordinates (x, y) in the L. Currently, there are various algorithms to calculate initial matching costs. Here, we employ AD-Gradient to calculate matching costs, which is the absolute difference of color intensity and gradient. The AD-Gradient can be defined by Eq. (1). The larger matching cost, the lower the similarity between the two pixels.

$$C_d(p) = \beta \cdot \min\left(\frac{1}{M}\left\|L(p) - R(p_d)\right\|_1, \tau_1\right) + (1-\beta)\cdot \min\left(\nabla_x L(p) - \nabla_x R(p_d), \tau_2\right), \quad (1)$$

where M represents the number of image channels, e. g. M is 3 for RGB images and 1 for gray images. $L(P)$ represents the color vector of pixel p in the L. ∇_x represents the gradient in the x-direction. p_d represents the pixel in the R corresponding to the p with disparity d. Its coordinates can be expressed as $p_d = (x-d, y)$. $R(p_d)$ represents the color vector of the p_d in the R. β is used to control the weight of color density and gradient. Here we set β to 0.11 according to the optimal values in Yang's paper. τ_1 and τ_2 are the thresholds for color intensity and gradient respectively in order to reduce the effect of noise and occluded pixels. They are set to 7 and 2 respectively. Calculating the matching cost of each pixel in the left view with each possible disparity value can obtain a 3D cost volume, and the points in this volume are matching costs of each pixel at each disparity level.

Below, we describe the proposed mPMST. This method in calculating aggregated costs maintains a linear computational complexity. It enables pixels to receive support from similar pixels in more directions than original method based on MST. We treat the reference image as an eight-connected undirected graph $G = (V, E)$ as shown in the Figure 4 (a). Here, E and V denotes all connected edges between neighboring pixels and all image pixels respectively. The edge weight between a pair of neighboring pixels can be defined as $w(p, q)$. It can be expressed as follows:

$$w(p, q) = w(q, p) = \max_{m\in[1,M]} |R_m(p) - R_m(q)|, \quad (2)$$

Here, m represents a color vector among M image channels. Using the max color difference can ensure that two similar adjacent pixels transmit information with a high weight to each other. There is a basic assumption in the research field of stereo matching: similar pixels in neighboring pixels usually have similar disparity. Using the maximum value instead of other possibilities (such as the average value) is an empirical choice and has better robustness. If two neighboring pixels are in a smooth region, the color difference between them is small, and vice versa. This can explain why the edges of the minimum spanning tree should not cross depth discontinuous regions as much as possible.

Finally, the Kruskal algorithm is used to establish a mPMST shown in Figure 4 (b). The distance between two arbitrary nodes on the tree is the sum of the minimum edge weights connecting the two nodes. Assume that any two pixels are denoted by s and t respectively, the distance between them can be expressed as $S(s,t) = S(t,s)$. Its calculation equation can be expressed as follows:

$$S(s,t) = S(t,s) = \exp\left(-\frac{w(t,s)}{\sigma}\right), \tag{3}$$

where σ is a parameter used to control the support strength.

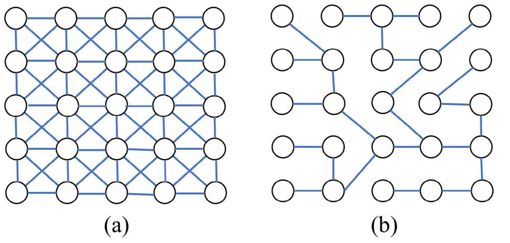

FIGURE 4. The tree structure of our method. (a) Eight-connected undirected graph of a reference image. (b)The structure of mPMST.

MST can ensure that the matching cost of each pixel is only aggregated on similar neighboring pixels. This characteristic derives from the definition of edge weights, as the total weight of MST is less than or equal to the total weight of all other possible spanning trees. Our mPMST increases the paths to select for cost aggregation, resulting in overall smaller total weights than MST and each pixel can more accurately obtain support information from other pixels.

After the tree structure is established, the aggregated cost can be calculated. We set $C_d(p)$ as the matching cost of pixel p in the image R with disparity d. $C_d^A(p)$ denotes the final aggregated cost, which can be expressed by the formula as follows:

$$C_d^A(p) = \sum_{q \in S} S(p,q) \cdot C_d(q) \tag{4}$$

Here, S denotes all pixels in the superpixel S. In the process of cost aggregation, each pixel can receive weighted support information from adjacent pixels within the same superpixel. However, directly using this definition formula to calculate aggregated costs is quite time-consuming. In order to reduce the time complexity of cost aggregation, we calculate the aggregated cost in two steps: cost aggregation from the root node to the leaf node and cost aggregation from the leaf node to the root node. These two steps are shown in Figure 5.

The first step is aggregating the matching cost from the leaf node to root node. If the temporal aggregated cost of the node v in this step is denoted as $C_d^{A\uparrow}$, its calculation formula

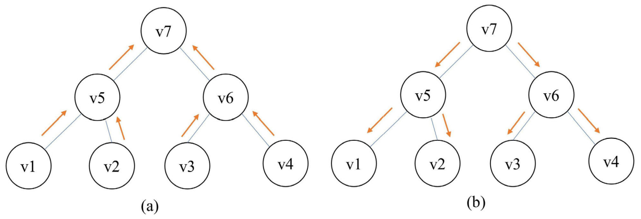

FIGURE 5. Two steps of cost aggregation. (a) From the leaf node to the root node. (b) From the tree root node to the leaf node.

can be expressed as follows:

$$C_d^{A\uparrow}(v) = C_d(v) + \sum_{P(v_i)=v} w(v,v_i) \cdot C_d^{A\uparrow}(v_i), \tag{5}$$

Here v_i represents the number of tree nodes, and $P(v_i) = v$ represents the parent nodes of v_i. Especially, the aggregated costs of leaf nodes are still equal to their initial matching costs in this step.

The second step is cost aggregation from the root node to leaf node. The final aggregated cost of each pixel can be calculated using the temporal results in the first step, and the final aggregated cost $C_d^A(v_i)$ can be represented as follows:

$$\begin{aligned} C_d^A(v_i) &= w(P(v_i), v_i)\, C_d^A(P(v_i)) \\ &\quad + \left[1 - w^2(v_i, P(v_i))\right] C_d^{A\uparrow}(v_i), \end{aligned} \tag{6}$$

By using this computation approach, cost aggregation has a linear time complexity in computation, because the distance between pixels can be accumulated by tracking from the leaf node to the root node. They only need to perform two addition or subtraction and three multiplication calculations. Because the mPMST in this section is run inside each superpixel, we can name it inside-superpixel-level mPMST. The aggregated costs at this level can be named inside-superpixel-level aggregated costs.

B. SUPERPIXEL-LEVEL MPMST

In order to improve the matching accuracy of low-texture regions, it is essential to construct a mPMST among all superpixels. Each superpixel is treated as a tree node, and the edge weights are determined by the similarity between adjacent superpixels. Let the matching cost of superpixel S with disparity d be denoted as $C_d(S)$, where S corresponds to a set of pixels in the reference image, and $|S|$ represents the number of pixels contained in the superpixel S, $C_d(S)$ can be expressed as follows:

$$C_d(S) = \sum_{p \in S} C_d(p) \Big/ |S|, \tag{7}$$

Following the superpixel segmentation, the resulting superpixel image does not form a regular grid. We establish edges between each pair of neighboring superpixels. Each edge weight between two superpixels is computed via the color distribution differences between them. Finally, a graph $G_R = (V_R, E_R)$ with uncertain connectivity numbers was formed as shown in Figure 6. V_R represents the set of all superpixels and E_R donates all edges.

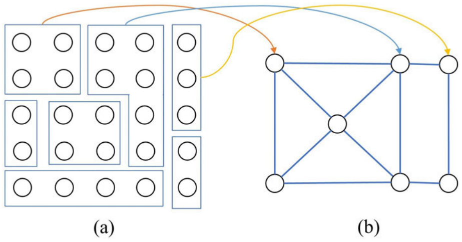

FIGURE 6. The process of building superpixel-level mPMST. (a) The result of SLIC. (b) The tree structure build by superpixels.

There are various metrics that can be used to define the distance between two adjacent superpixels. We calculate color histograms and utilize the difference between the primary colors (also known as patterns) for each superpixel to define the distance. This metric is computationally simple and more robust than the average color difference, as SLIC sometimes generates fragments that slightly cross different color regions. For any two neighboring superpixels V_i and V_j, the edge weight between them $w\left(V_i, V_j\right)$ can be calculated by the following formula:

$$w\left(V_i, V_j\right) = \left|R_{V_i} - R_{V_j}\right|, \tag{8}$$

Here, R_{V_i} and R_{V_j} represent the primary colors of the two superpixels respectively. Similar to Eq. (3) in the previous section, the distance between two superpixels is equal to the sum of all minimum connected edge weights between them, which can be represented by $S\left(V_i, V_j\right) = S\left(V_j, V_i\right)$.

$$S\left(V_i, V_j\right) = S\left(V_j, V_i\right) = \exp\left(-\frac{w\left(V_i, V_j\right)}{\sigma}\right) \tag{9}$$

Finally, we can establish a minimum spanning tree using the Kruskal algorithm. Assuming the aggregated cost of superpixel V_i to be $C_d^A\left(V_i\right)$, the calculation equation is as follows:

$$C_d^A\left(V_i\right) = \sum_{V_j \in V} S\left(V_i, V_j\right) \cdot C_d\left(V_i\right) \tag{10}$$

This cost aggregation method is the same as described in the previous section. After completing cost aggregation, each pixel can obtain a superpixel-level aggregated cost and an inside-superpixel-level aggregated cost. The next section will solve how to fuse the two aggregated costs.

C. ADAPTIVE WEIGHT FUSION

In this section, we solve how to fuse the two aggregated costs for each pixel. Figure 7 illustrates the weighted contribution of all pixels in the superpixel to the red test point, where the row 1 (a) and (b) are the original image and a close-up region with the red test point respective, and the row 2 are the weighted contribution of all superpixels to the test region at superpixel-level mPMST and the weighted contribution of pixels inside superpixel to the test point at inside-superpixel-level mPMST respectively. For low-texture regions, the superpixel-level mPMST has a better support effect, which can preserve edges based on SLIC superpixel segmentation and propagate support information to other low-texture regions. Thus, the superpixel-level mPMST should dominate in low-texture regions. The mPMST inside each superpixel has a good effect in high-texture regions, where each pixel can accurately obtain support from other neighboring similar pixels, leading to high matching accuracy.

Based on this characteristic, we introduce the concept of image entropy to propose a novel adaptive weight fusion method.

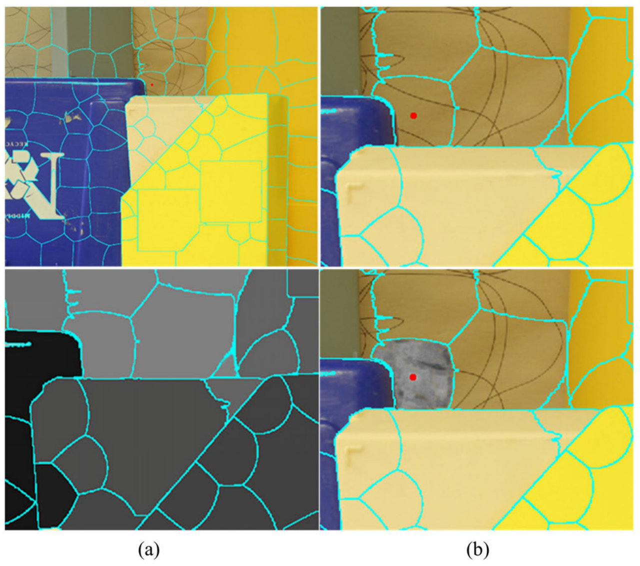

FIGURE 7. Support weights of two levels mPMST to the red test point and test region. A larger grayscale value has a greater contribution.

Image entropy is a statistical representation of image features, providing valuable information to describe an image. If all pixels in an image have the same gray level or the same color intensity, the entropy of the image is minimized. Conversely, if each pixel in an image has a unique gray level or color intensity, this image has maximum entropy. Because complex textures tend to result in scattered distribution of gray level or color intensity, the image entropy can be used to quantify texture complexity. Define the one-dimensional grayscale entropy of a grayscale image in superpixel S as follows:

$$E\left(S\right) = -\sum_{i=0}^{255} P_R\left(i\right)\ln P_R\left(i\right) - \sum_{i=0}^{255} P_G\left(i\right)\ln P_G\left(i\right) - \sum_{i=0}^{255} P_B\left(i\right)\ln P_B\left(i\right), \tag{11}$$

Here, i represents the value of color intensity, which can vary from 0 to 255. $P_R\left(i\right)$ represents the distribution probability of color intensity in the superpixel S in the R color channel. This distribution probability can be obtained by counting the number of pixels with a given value i and then divided by the total number of pixels. According to the Eq. (11), we can see that the range of entropy values is from 0 to 16.6355.

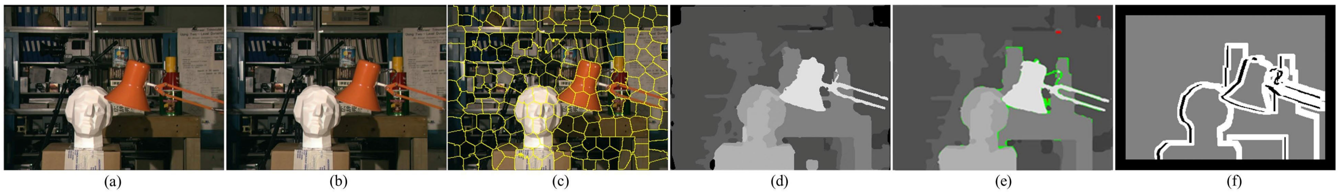

FIGURE 8. The results of our method in Tsukuba. (a) The right view. (b) The left view (reference image). (c) The result of SLIC. (d) The initial disparity map. (e) The disparity refined by our method. (f) The division of test regions.

In this way, we can compute the adaptive weight λ_S for superpixels S.

$$\lambda_S = \frac{16.6355 - E(S)}{16.6355}, \tag{12}$$

We denote the final aggregated cost of pixel p with disparity d as $C_d^{A'}(p)$. Since superpixel S includes pixel p, we have $p \in S$. $C_d^{A'}(p)$ can be expressed as follows:

$$C_d^{A'}(p) = \left(1 - \lambda_S\right) C_d^A(p) + \lambda_S C_d^A(S), \tag{13}$$

In this way, we obtain the final aggregated cost for each pixel with all possible disparity values. Finally, the disparity map is generated by using WTA algorithm. The final aggregated cost usually does not require normalization processing, as WTA algorithm directly selects the disparity value with the minimum final aggregated cost for each pixel as the final disparity value. WTA algorithm can be expressed as follows:

$$D(p) = \arg\min_d \left(C_d^{A'}(p)\right), \tag{14}$$

where $D(p)$ denotes the disparity value of pixel p.

Overall, the novel cost aggregation method proposed in this paper still belongs to the framework of non-local cost aggregation. It assimilates both local information and information from other non-local pixels.

D. DISPARITY MAP REFINEMENT

In this section, we propose a novel disparity map refinement method using the cost aggregation structure proposed in this paper. Refining disparity maps is usually based on Left-Right Consistency (LRC) test. Firstly, we distinguish between stable and unstable points through LRC test. The specific approach is: we use the stereo matching algorithm proposed in this paper to generate two disparity maps for the left and right views in a stereo image pair respectively. Secondly, if the disparity value of pixel p in the left view is $D(p)$, the horizontal coordinate of the corresponding pixel p' in the right view is $p - D(p)$. The vertical coordinate is the same as pixel p. Assuming that the disparity map based on the right view has a disparity $D(p')$ in pixel p', if $\left|D(p) - D(p')\right| \neq 0$, then this pixel is considered an unstable pixel, otherwise it is considered a stable point. Therefore, we can set a new cost $C_{new}^d(p)$ for each pixel as follows:

$$C_d^{new}(p) = \begin{cases} |d - D(p)| & p \text{ is stable and } D(p) > 0, \\ 0 & else, \end{cases} \tag{15}$$

The aggregated cost of all unstable pixels at all disparity levels is set to 0, so they completely dependent on information from stable pixels. We respectively build mPMST inside each superpixel to aggregate costs. The aggregated costs of stable pixels can propagate to unstable pixels. For some superpixels without stable points, we run superpixel-level mPMST to aggregate cost. Finally, the WTA algorithm is used again to calculate the disparity value of unstable pixels. However, we found that there are often black holes near image boundaries. These regions typically are composed of the superpixels without stable pixels and serve as tree leaves on superpixel-level mPMST. In order to address this issue, we use the accurate disparity with the highest frequency of occurrence at the edge of each black hole to fill in.

This non-local disparity map refinement method can be more convenient than local refinement algorithms and achieve better refinement performance without specifying support window size.

IV. EXPERIMENTS AND ANALYSIS

In this section, the method proposed in this paper is compared with the state-of-the-art stereo matching methods including GF [22], MST [24], [25], ST [26], CROSS-E [27] and EMST [18]. We use Middlebury and KITTI 2015 datasets to evaluate them. We compared the disparity maps obtained by various methods with real disparity maps. The algorithm accuracy is evaluated by the error rate in non-occluded regions, where the error rate is the percentage of erroneous pixels having disparity error larger than 1 and 3 pixels in Middlebury and KITTI 2015 datasets respectively. For the MST, GF, and ST, the author provided source codes using C++ or MATLAB language, so we can use them directly. For CROSS-E and EMST, we implement their codes by using C++ language. Their parameters are all set to the optimal parameters in their papers. For all the datasets, the parameter of our method is constant: $\sigma = 0.05$. σ is set based on previous research experience. The experiment environment is my personal laptop, equipped with a CPU of Intel Core i5 8400 and system of Windows 10. No any acceleration approach is used in these experiments.

A. MIDDLEBURY BENCHMARK

The Middlebury [35] datasets provide a large number of stereo image pairs for indoor scenes. We first use four classic image pairs to test: Tsukuba, Venus, Teddy, and Cones, using the disparity map refinement method proposed in this paper.

Other algorithms for comparison use their own disparity map refinement method. These four images provide a range of discontinuous regions, so we evaluate the matching accuracy of methods in non-occluded regions ("non"), all pixels ("all"), and discontinuous regions ("disc"). There are a total of 26 stereo image pairs in Middlebury 2006, which have more complex scenes than the four classic images mentioned above, greatly improving the difficulty of matching. Among these datasets, Midd1, Midd2, Monopoly, and Plastic have a large number of low-texture regions and textureless regions, which is unfair to local and non-local algorithms. Therefore, these four datasets are removed. The number of superpixels is set to 180.

As illustrated in Table 1, our method achieved an average error rate of 5.21%, which is the lowest among these methods. However, in Teddy and Cones, our method in discontinuous regions is not better than others. Thanks to our disparity map refinement, we can fill black hollow regions. Therefore, our method has a highest average accuracy in all pixels ("all"). Figure 9 visually presents the disparity result of various algorithms, where erroneous pixels in non-occluded regions are marked in red and erroneous pixels in discontinuous regions are marked in green. From Figure 9, it can be seen that our method has far fewer erroneous pixels than other methods and can preserve edges to some extent, especially on the top of teddy bear (highlighted in a yellow box). There are also no black holes in our disparity maps.

Next, we use more complex image pairs from Middlebury 2006 to test the performance of cost aggregation without using any post-processing techniques. This experiment does not include Tsukuba, Venus, Teddy, and Cones which have already been tested. We use the error rate of erroneous pixels in non-occluded regions to evaluate the accuracy of various methods. In order to accelerate the experiment speed and accurately reflect the advantages and disadvantages of these methods, we utilize half-size images on the Middlebury datasets. The quantitative evaluation results are presented in Table 2. The Figure 11 shows the disparity results of various methods, where erroneous pixels in non-occluded regions are marked in red.

From the Table 2, it can be seen that our method produces the competitive results of the evaluated methods on these datasets and gets the best performance on 18 datasets. Our average error rate is 3.32% lower than MST and 0.63% lower than EMST. As shown in Figure 11, our method not only makes the edges of objects clearer than other algorithms, but also improves accuracy of low-texture regions. On Baby1 and Baby2 image pairs, the baby's hands and head are more easily recognizable, which are highlighted in yellow boxes. The object under the baby is a large low-texture region where our approach still has the fewest erroneous pixels.

Finally, we test these methods on Middlebury 2014. As an upgrade version of Middlebury 2006, it contains 20 high-resolution image pairs with different scenes. We test our method at quarter-resolution images for fair comparison with other state-of-the-art methods.

For a quantitative comparison, Table 3 lists the error rates of various methods in non-occluded regions. Our method obtains the lowest average error rate compared with others and 8.53% higher than testing on Middlebury 2006 datasets. Additionally, our method outperforms the other five algorithms in 11 image pairs. The error rate of our method is 1.14% lower than the second ranked algorithm. All the methods perform poorly on Shelves, Vint and Jade. This is because these datasets have a wide range of disparity variations. For visual comparison, the disparity maps obtained by using the six methods are shown in Figure 12. From Figure 12, it is obvious that our method can produce fewer erroneous pixels in low-texture and duplicate-texture regions, especially in the motor, garbage can and ground regions within black boxes.

Below, we analyze the reasons for the experimental results. The GF algorithm performs best among local cost aggregation algorithms, but it is limited to cost aggregation within a support window and does not perform well in low-texture regions. Its effect is superior to MST according to Table 2 and Table 3. Other algorithms belong to non-local algorithms and adopt different tree structures for cost aggregation. The MST and ST establish a minimum spanning tree on the entire graph and propagate the cost from only four directions, which makes the algorithm not only sensitive to noise. The ST algorithm also relies heavily on the merging results of several minimum spanning trees. However, it has lower accuracy than MST in four Middlebury 2014 datasets, such as Motor, Piano, Recycle and PlaytP. Because the colors and graphics in these image pairs are more complex, it is difficult to merge several trees appropriately. The CROSS-E uses an improved cross tree structure for cost aggregation, but it is prone to edge blurring and overly relies on edge extraction. Its average error rate is lower than MST and ST on Middlebury 2006 and Middlebury 2014, because utilizes edge information from Canny and SLIC. The EMST utilizes Canny edges to change edge weights, which can preserve edges. However, its accuracy improvement in low-texture regions is small, such as Baby1 and Baby2 in Figure 10. Its average error rate is 0.55% lower than CROSS-E in the Middlebury 2014 datasets. Our method utilizes SLIC to obtain accurate edge information, which can make depth boundaries and the shape of objects clearer. We use mPMST at superpixel level to provide support information for low-texture regions. The mPMST at the inside-superpixel level can effectively resist noise and improve the accuracy in high-texture regions, because similar pixels usually have similar disparity values. Finally, we make use of adaptive weights to merge the two aggregated costs. Our method overcomes the shortcomings of currently existing algorithms and has good results in both low-texture regions and high-texture regions.

B. KITTI 2015 BENCHMARK

The KITTI 2015 datasets are made up of real-world scenes captured by mobile vehicles [34]. This dataset contains a lot of stereo pairs with a large portion of textureless regions, such as roads and walls. In order to evaluate the robustness and accuracy of the proposed method, we run our method on some classical KITTI 2015 datasets to conduct a further comparison with other state-of-the-art methods. For convenience, we only use 100 pairs of stereo image pairs in the training set to evaluate the methods. Due to the large image size of the KITTI 2015 datasets, we set the number of superpixels to 400. The testing area is divided into all pixels and non-occluded regions, represented by "all" and "non" respectively. We evaluate the accuracy of these methods by

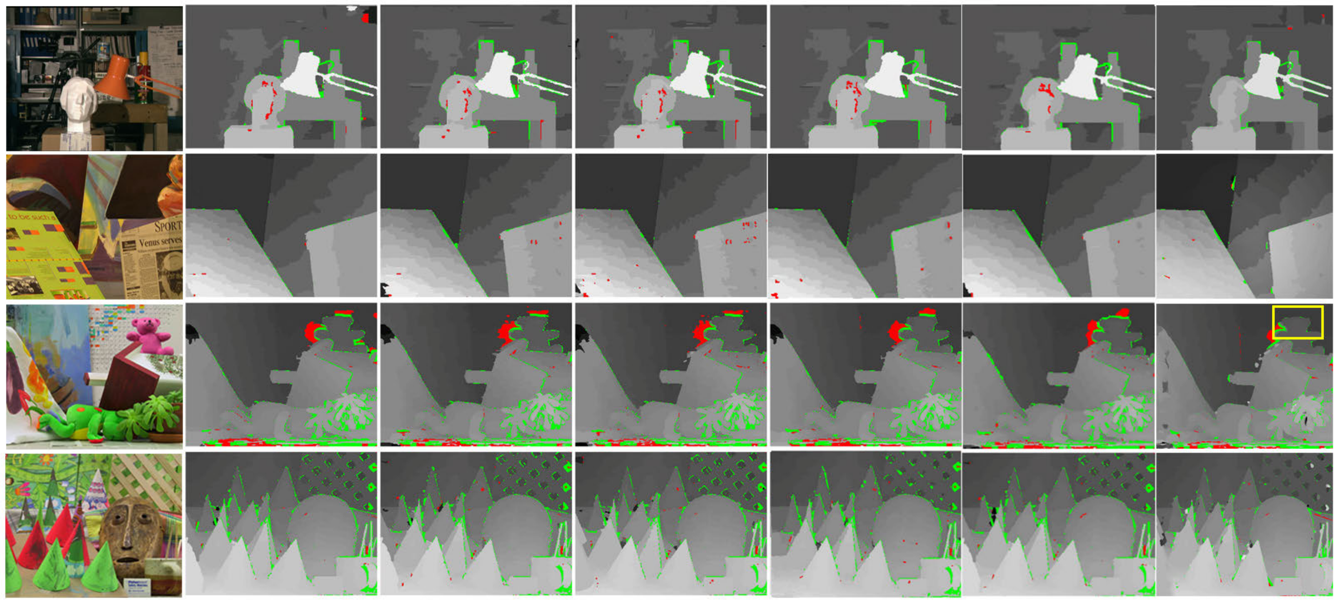

FIGURE 9. The refined disparity results in four classical Middlebury datasets. From left to right: Left view (reference image), GF, MST, ST, CROSS-E, EMST, Our method. From top to bottom: Tsukuba, Venus, Teddy, Cones.

TABLE 1. Comparison results of various methods in Tsukuba, Venus, Teddy and Cones.

Methods	Tsukuba			Venus			Teddy			Cones			Average
	non	all	disc	non	all	disc	non	all	disc	non	all	disc	
GF	1.51	1.85	7.61	0.20	0.39	2.42	6.16	11.62	16.0	2.71	**8.24**	**7.65**	5.55
MST	1.47	1.85	7.88	0.25	0.42	0.42	6.01	11.69	**14.3**	2.87	8.45	8.10	5.48
ST	1.25	1.68	6.69	0.20	0.30	1.77	6.00	11.95	15.0	2.77	8.82	7.81	5.35
CROSS-E	1.68	1.99	7.82	0.22	0.39	2.42	6.16	11.81	16.0	2.71	**8.24**	7.66	5.44
EMST	1.16	**1.35**	5.94	0.18	0.37	**0.32**	5.56	10.93	13.5	2.75	8.33	7.87	5.24
OUR	**0.57**	1.55	**4.84**	**0.12**	**0.25**	2.21	**4.18**	**10.50**	17.7	**2.05**	8.55	9.85	**5.21**

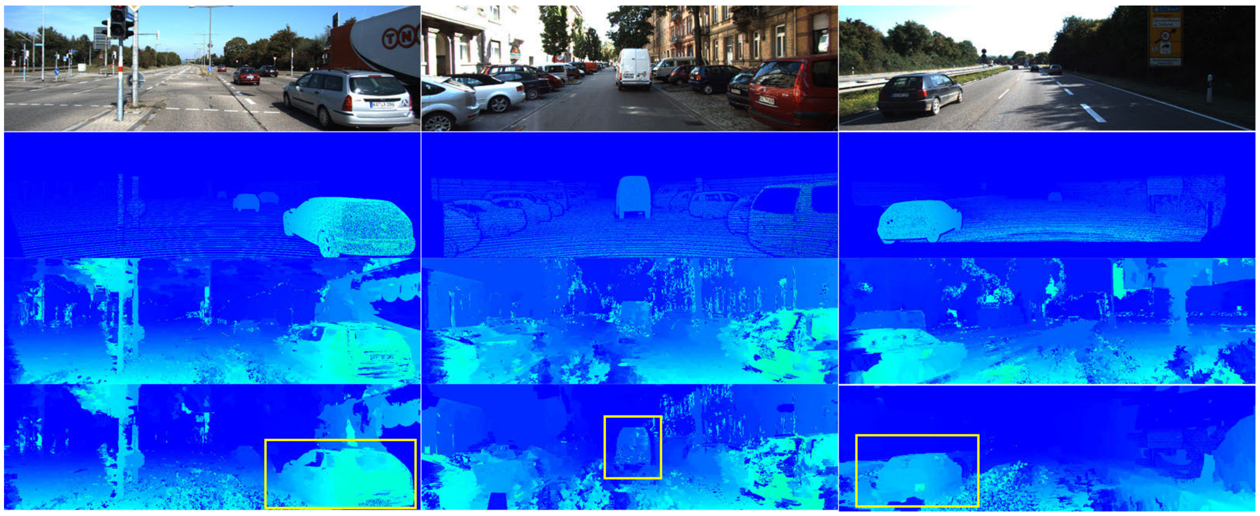

FIGURE 10. The disparity results on KITTI 2015 datasets. From top to bottom: the left view (reference image), real disparity map, MST, our method. From left to right: 000049_10, 000004_10, 000100_10.

TABLE 2. Error percentages in non-occluded regions of various algorithms on Middlebury datasets.

Datasets	GF	MST	ST	CROSS-E	EMST	Our
Aloe	5.43	6.34	4.55	3.52	**3.14**	4.66
Art	11.90	13.62	11.47	**8.58**	11.31	9.12
baby1	4.93	8.07	5.54	4.63	5.21	**3.01**
baby2	**5.92**	18.27	15.20	10.73	12.37	6.33
baby3	6.69	5.71	5.07	5.27	3.95	**3.41**
Books	15.04	12.03	10.33	8.02	**7.52**	10.06
Bowling1	19.60	23.95	15.8	15.6	16.32	**13.26**
Bowling2	8.29	12.85	9.25	8.83	8.51	**8.15**
Cloth1	0.78	0.73	0.71	**0.15**	0.46	0.23
Cloth2	2.86	4.75	2.90	1.84	2.15	**1.72**
Cloth3	1.71	2.57	2.26	1.53	3.11	**1.39**
Cloth4	1.45	1.92	1.66	**0.77**	1.39	0.83
Dolls	8.53	9.81	8.48	7.11	7.63	**6.11**
Lampshade1	13.47	11.57	10.41	10.45	**7.53**	10.42
Lampshade2	15.32	13.56	21.74	14.52	13.91	**13.76**
Laundry	18.75	13.56	13.83	13.71	**10.21**	12.87
Moebius	9.28	8.54	8.16	7.89	7.65	**7.52**
Flowerpots	13.09	19.57	15.69	15.48	13.78	**11.12**
Wood1	4.22	10.56	9.51	4.19	4.96	**3.58**
Wood2	3.78	1.38	2.01	**0.83**	1.03	1.16
Reindeer	9.84	9.15	7.13	5.59	5.64	**5.31**
Rock1	2.96	2.60	2.41	1.65	1.55	**1.34**
Rock2	1.92	1.98	1.75	1.45	1.67	**1.31**
average	8.09	9.26	8.08	6.62	6.57	**5.94**

TABLE 3. Error rates in non-occluded regions of methods on Middlebury 2014 datasets.

Methods	Adir	ArtL	Jade	Motor	MotorE	Piano	Playr	Pipes	Playt	PlaytP	Recycle	Shelves	Teddy	Vint	Avg
GF	18.44	13.44	21.22	7.52	16.94	19.22	21.27	11.09	20.38	11.25	9.46	36.31	6.82	26.97	17.17
MST	17.99	16.75	25.12	9.94	20.93	20.64	18.13	12.68	21.78	14.81	7.76	31.59	5.68	30.16	18.14
ST	19.34	12.53	23.71	11.45	23.36	21.20	18.45	12.04	19.89	16.82	10.48	31.05	5.35	26.28	18.00
CROSS-E	17.68	**10.03**	20.38	8.95	16.12	18.34	16.67	10.58	18.93	11.59	7.32	28.21	4.73	24.35	15.28
EMST	**17.24**	11.38	22.74	8.68	17.66	**17.56**	17.87	11.65	17.65	12.36	7.64	30.53	4.95	**23.61**	15.82
Our	17.77	10.39	**18.51**	**7.75**	**15.53**	18.63	**15.38**	**10.17**	**16.21**	**10.67**	**6.91**	**27.55**	**4.16**	25.88	**14.68**

TABLE 4. Error rates of various methods in non-occluded and all pixels for ablation studies.

Datasets	MST		MST + our refinement		ST		ST + our refinement		mPMST		Two-level mPMST		Two-level mPMST + our refinement	
	non	all	non	all	non	all	non	all	non	all	non	all	non	all
Art	13.62	22.32	13.44	19.47	11.47	22.24	10.63	20.77	10.74	21.94	9.12	20.67	8.56	19.85
Aloe	6.34	10.52	6.03	9.31	4.55	9.87	6.18	8.16	4.72	9.17	4.66	8.94	4.07	7.73
Baby1	8.07	16.22	7.54	15.63	5.54	15.82	4.94	14.82	5.13	14.62	3.01	13.22	3.01	12.19
Baby3	5.71	14.65	5.12	13.26	5.07	15.19	5.26	13.26	4.88	14.13	3.41	14.13	3.16	12.82
Dolls	9.81	21.22	9.53	18.91	8.84	20.44	8.61	17.52	8.49	18.68	6.11	17.94	5.72	15.46
Teddy	7.31	11.60	6.12	9.85	6.00	11.03	5.58	10.17	4.56	10.69	4.34	10.92	4.18	10.50
Wood1	10.56	24.12	10.47	21.97	9.51	22.46	8.78	21.13	9.23	20.54	3.58	19.71	2.93	17.13
average	8.77	17.24	8.32	15.49	7.28	16.75	7.14	15.12	6.82	15.68	4.89	15.08	4.52	13.67

average error rate in test regions. The testing results of various methods is displayed in Table 3 and the representative rendering is in Figure 10.

From the Table 5, it can be seen that our method performs better than others. Because it incorporates SLIC superpixel segmentation, resulting in better preservation of the car

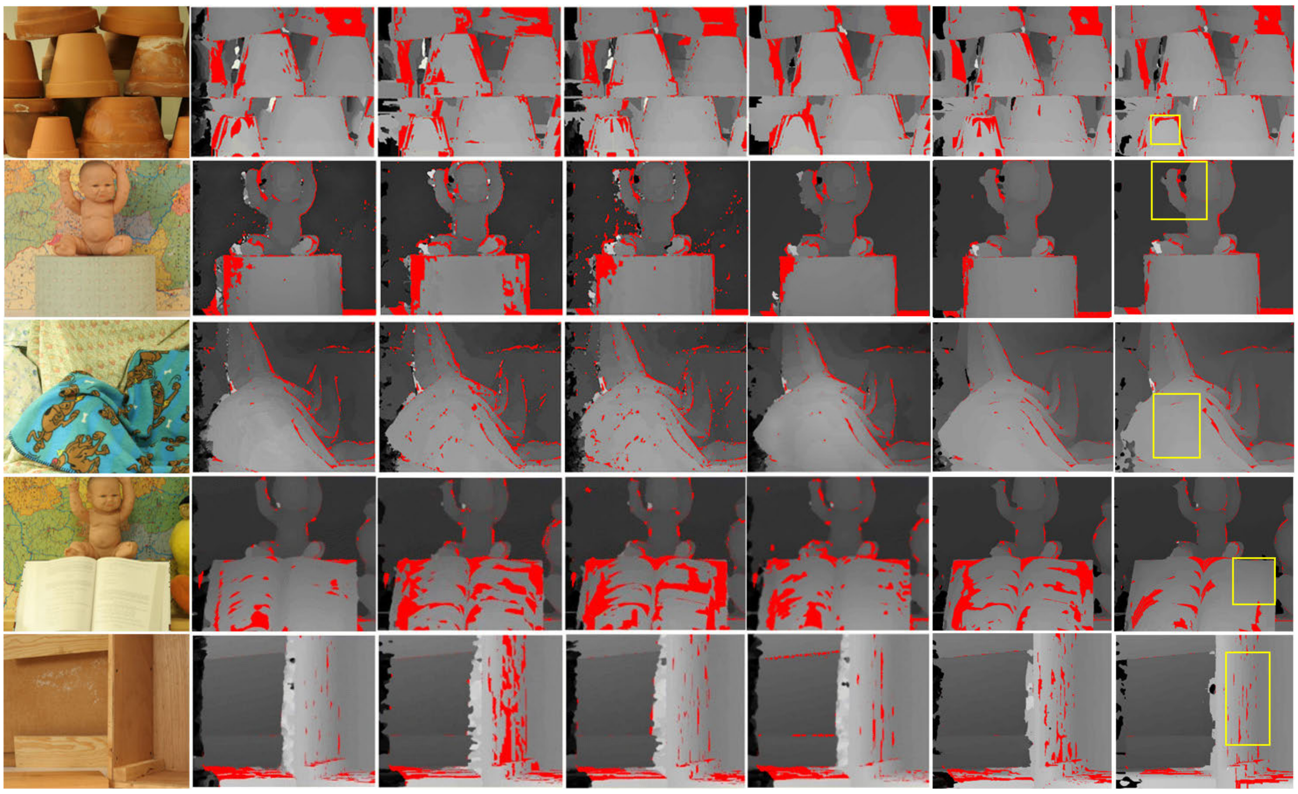

FIGURE 11. The disparity results without post-processing techniques on Middlebury 2006 datasets. From left to right: the left view, GF, MST, ST, CROSS-E, EMST, Our method. From top to bottom: Flowerpots, Baby1, Cloth3, Baby2, Wood1.

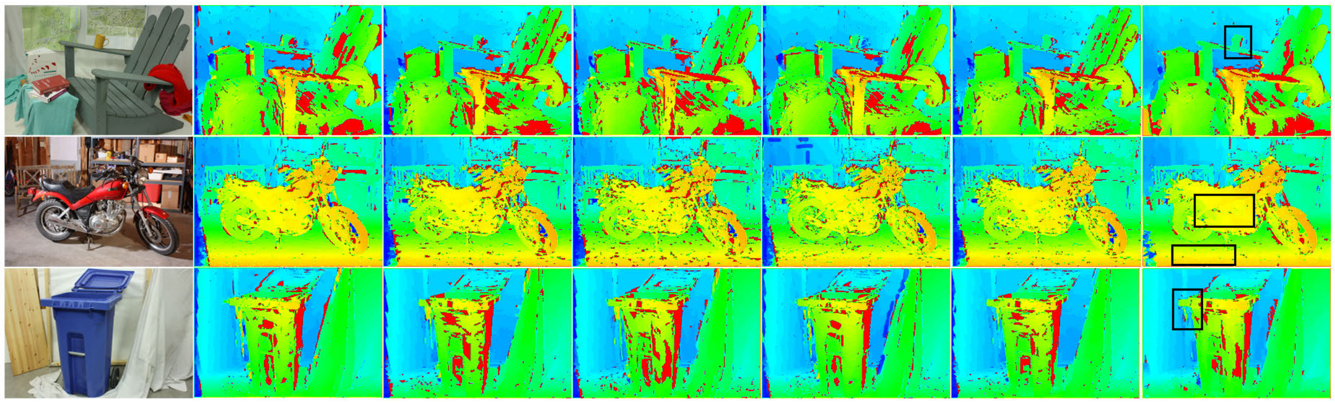

FIGURE 12. The disparity results without post-processing techniques on Middlebury 2014 datasets. From left to right: the left view, GF, MST, ST, CROSS-E, EMST, Our method. From top to bottom: Adirondack, Motorcycle, Recycle.

shape, such as regions within yellow boxes in Figure 10. In background regions, due to the cost aggregation at superpixel level, there are fewer erroneous pixels. MST and ST perform poorly on the KITTI 2015 datasets mainly due to their sensitivity to noise and poor performance in low-texture regions. Overall, the KITTI 2015 datasets present a significant challenge for both local and non-local algorithms.

C. RUNNING TIME COMPARISON

We compare the average running time of our method with the five state-of-the-art methods on four standard Middlebury datasets. The experiment is on a PC platform with a 2.80 GHz Intel CPU and 8G memory. All the methods are implemented in C++.

The average running time of GF, MST, ST, CROSS-E, EMST, and ours are GF-15.82 s, MST-5.31 s, ST-6.23 s, CROSS-E-5.74 s, EMST-5.42 s and ours-5.81 s respectively. Our method is faster than GF and ST, but slower than MST and EMST. For GF, although it is independent of the kernel size, it still needs to invert a 3 * 3 symmetric matrix for each pixel, which takes a lot of time. Therefore it is the slowest method among these methods. For MST, this method spends the most of time in constructing a minimum spanning tree. Therefore, it is a computational burden to build a tree for

TABLE 5. Average error percentages in non-occluded regions of different methods on KITTI2015 datasets.

Methods	all	non
GF	29.21	25.23
MST	35.28	31.56
ST	33.82	30.44
CROSS-E	28.51	25.14
EMST	27.36	26.82
Our	**26.43**	**24.51**

high-resolution images. Its cost aggression process has linear complexity, so spend a little time in this process. The EMST is similar with MST. It takes a portion of time to segment images and extract edges. It is slightly slower than MST. For ST, it requires a unique minimum tree structure and an initial estimation of the disparity map, which makes it slower than other tree methods. For CROSS-E, its tree structure is fixed and does not require a lot of time to build trees. However, it needs more time to finish computing aggregated costs. In addition, The CROSS-E requires SLIC and Canny to obtain edge information. Therefore, it spends more time than MST and EMST. For our method, it spends the most of time on building trees because needs to build trees by selecting more paths. This method builds a tree inside each superpixel, which is much faster than in an entire image. This feature can save a lot of time to compensate for the time consumption caused by more paths. Thus, the running time of our method is between ST and CROSS-E. Besides, our method can be greatly accelerated through programming techniques such as multithreading, because building a tree inside each superpixel is independent of other superpixels. This can serve as future work.

D. ABLATION STUDIES

In this section, ablation studies are performed to verify the effectiveness of mPMST, two-level cost aggregation, entropy-based adaptive weight fusion and our refinement method. We select 7 representative stereo image pairs from Middlebury datasets to test the impact of each part on accuracy. The results are listed in Table 4.

mPMST: mPMST can optimize cost aggregation paths based on MST, achieving more effective cost aggregation. According to Table 4, the average error rate of mPMST is 1.92% lower than MST in non-occluded regions, and 2.92% lower in all pixels. The mPMST has a significant improvement in high-texture images such as Teddy, but has a little improvement in images containing a lot of low-texture regions such as Wood1.

Two-level cost aggregation: In order to further improve accuracy, especially in low texture areas, we need two levels of cost aggregation: inside-superpixel and superpixel levels. Each similar low-texture superpixel can provide support information to each other with high weights. According to Table 4, the average error rate of two-level mPMST is 1.93% lower than mPMST in non-occluded regions, and 0.6% lower in all pixels. The improvement in accuracy is mainly in image pairs with a lot of low-texture regions, such as Wood1 and Baby1.

Refinement method: Our refinement method can refine initial disparity maps. We apply this refinement method to refine the initial disparity maps of MST, ST, mPMST, respectively. The results are quantitatively presented in Table 4. On the initial disparity maps generated by these methods, the accuracy can be improved by 1% to 2% in all pixels. There our refinement method is effective.

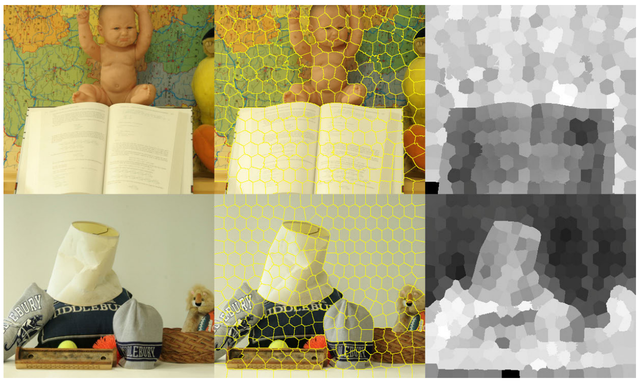
FIGURE 13. The relative size of entropy for each superpixel in Baby2 and Widd1.

Entropy-based adaptive weight fusion: This weight can control the proportion of aggregated values between two levels. According to Eq. (11), entropy can measure texture complexity. We show the relative size of entropy for each superpixel in Figure 13, where the larger the grayscale value, the more complex the texture is. From Figure 13, we can see that entropy can effectively distinguish between low-texture and high-texture regions and quantify them. Therefore, our fusion method is suitable for fusing aggregated costs at two levels.

V. CONCLUSION

In this paper, we propose a novel structure of multi-path minimum spanning tree with two levels: inside-superpixel level and superpixel level, where the two cost aggregation values are combined using a novel adaptive weight based on image entropy. The average error rate of 14.68% is obtained on Middlebury 2014 datasets. Our method can achieve better performance than other state-of-the-art methods and make edges clearer. For our disparity map refinement method, the accuracy in all pixels can be improved by between 1% and 2% in an initial disparity map.

However, SLIC cannot maintain the boundary for long and thin features and sometimes can cross object boundaries. These issues limit further improvement in accuracy. We can choose a more advanced superpixel segmentation algorithm in the future. The running speed of our method can be greatly improved by technologies such as multithreading and parallel computing. We leave these work for future studies.

ACKNOWLEDGMENT

The author expresses his sincere gratitude to the journal's editor and anonymous reviewers for their help in revising the paper.

REFERENCES

[1] H.-W. Chae, J.-H. Choi, and J.-B. Song, ''Robust and autonomous stereo visual-inertial navigation for non-holonomic mobile robots,'' *IEEE Trans. Veh. Technol.*, vol. 69, no. 9, pp. 9613–9623, Sep. 2020.

[2] C. Liu, W. Wei, B. Liang, X. Liu, W. Shang, and J. Li, ''ConvMLP-mixer based real-time stereo matching network towards autonomous driving,'' *IEEE Trans. Veh. Technol.*, vol. 72, no. 2, pp. 2581–2586, Feb. 2023.

[3] H. Luo, M. Wang, P. K.-Y. Wong, and J. C. P. Cheng, ''Full body pose estimation of construction equipment using computer vision and deep learning techniques,'' *Autom. Construct.*, vol. 110, Feb. 2020, Art. no. 103016.

[4] A. D. Pon, J. Ku, C. Li, and S. L. Waslander, ''Object-centric stereo matching for 3D object detection,'' in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2020, pp. 8383–8389.

[5] R. Fan, J. Jiao, J. Pan, H. Huang, S. Shen, and M. Liu, ''Real-time dense stereo embedded in a UAV for road inspection,'' in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jun. 2019, pp. 535–543.

[6] J. Salmerón-Garcióa, P. Ìñigo-Blasco, and D. Cagigas-Muñiz, ''A tradeoff analysis of a cloud-based robot navigation assistant using stereo image processing,'' *IEEE Trans. Autom. Sci. Eng.*, vol. 12, no. 2, pp. 444–454, Apr. 2015.

[7] C. Hui, R. Wang, X. Wen, J. Zhao, W. Chen, and X. Zhang, ''A novel recognition algorithm in 3D point clouds based for on local spherical harmonics,'' in *Proc. IEEE Int. Conf. Mechatronics Autom. (ICMA)*, Aug. 2019, pp. 1041–1046.

[8] B. Mallik, A. Sheikh-Akbari, and A.-L. Kor, ''HEVC based mixed-resolution stereo video codec,'' *IEEE Access*, vol. 6, pp. 52691–52702, 2018.

[9] D. Scharstein and R. Szeliski, ''A taxonomy and evaluation of dense two-frame stereo correspondence algorithms,'' *Int. J. Comput. Vis.*, vol. 427, pp. 7–42, Jan. 2002.

[10] O. Veksler, ''Stereo correspondence by dynamic programming on a tree,'' in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2005, pp. 384–390.

[11] M. Hallek, H. Boukamcha, A. Mtibaa, and M. Atri, ''Dynamic programming with adaptive and self-adjusting penalty for real-time accurate stereo matching,'' *J. Real-Time Image Process.*, vol. 19, no. 2, pp. 233–245, Apr. 2022.

[12] Q. Yang, L. Wang, and N. Ahuja, ''A constant-space belief propagation algorithm for stereo matching,'' in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2010, pp. 1458–1465.

[13] B. Lu, L. Sun, L. Yu, and X. Dong, ''An improved graph cut algorithm in stereo matching,'' *Displays*, vol. 69, Sep. 2021, Art. no. 102052.

[14] M. Chehaitly, A. Lalevee, T. Napoleon, and M. Jridi, ''High-speed and low-area implementations of a generic and parallel integral image architecture,'' in *Proc. 20th IEEE Interregional NEWCAS Conf. (NEWCAS)*, Jun. 2022, pp. 407–411.

[15] Z. Xie, S. Chen, and G. Orchard, ''Event-based stereo depth estimation using belief propagation,'' *Frontiers Neurosci.*, vol. 11, p. 535, Oct. 2017.

[16] C. Pan, Y. Liu, and D. Huang, ''Novel belief propagation algorithm for stereo matching with a robust cost computation,'' *IEEE Access*, vol. 7, pp. 29699–29708, 2019.

[17] C. Tomasi and R. Manduchi, ''Bilateral filtering for gray and color images,'' in *Proc. 6th Int. Conf. Comput. Vis.*, 1998, pp. 839–846.

[18] C. Zhang, C. He, Z. Chen, W. Liu, M. Li, and J. Wu, ''Edge-preserving stereo matching using minimum spanning tree,'' *IEEE Access*, vol. 7, pp. 177909–177921, 2019.

[19] D. Min, J. Lu, and M. N. Do, ''A revisit to cost aggregation in stereo matching: How far can we reduce its computational redundancy?'' in *Proc. Int. Conf. Comput. Vis.*, Nov. 2011, pp. 1567–1574.

[20] K. N. Chaudhury and S. D. Dabhade, ''Fast and provably accurate bilateral filtering,'' *IEEE Trans. Image Process.*, vol. 25, no. 6, pp. 2519–2528, Jun. 2016.

[21] R. G. Gavaskar and K. N. Chaudhury, ''Fast adaptive bilateral filtering,'' *IEEE Trans. Image Process.*, vol. 28, no. 2, pp. 779–790, Feb. 2019.

[22] A. Hosni, C. Rhemann, M. Bleyer, C. Rother, and M. Gelautz, ''Fast cost-volume filtering for visual correspondence and beyond,'' *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 2, pp. 504–511, Feb. 2013.

[23] K. He, J. Sun, and X. Tang, ''Guided image filtering,'' *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 6, pp. 1397–1409, Jun. 2013.

[24] Q. Yang, ''A non-local cost aggregation method for stereo matching,'' in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 1402–1409.

[25] Q. Yang, ''Stereo matching using tree filtering,'' *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 4, pp. 834–846, Apr. 2015.

[26] X. Mei, X. Sun, W. Dong, H. Wang, and X. Zhang, ''Segment-tree based cost aggregation for stereo matching,'' in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 313–320.

[27] F. Cheng, H. Zhang, M. Sun, and D. Yuan, ''Cross-trees, edge and superpixel priors-based cost aggregation for stereo matching,'' *Pattern Recognit.*, vol. 48, no. 7, pp. 2269–2278, Jul. 2015.

[28] C. Çigla and A. A. Alatan, ''Efficient edge-preserving stereo matching,'' in *Proc. IEEE Int. Conf. Comput. Vis. Workshops*, Nov. 2011, pp. 696–699.

[29] K. Zhang, J. Lu, and G. Lafruit, ''Cross-based local stereo matching using orthogonal integral images,'' *IEEE Trans. Circuits Syst. Video Technol.*, vol. 19, no. 7, pp. 1073–1079, Jul. 2009.

[30] L. Feng and K. Qin, ''Superpixel-based graph cuts for accurate stereo matching,'' *IOP Conf. Ser., Earth Environ. Sci.*, vol. 69, no. 1, Jun. 2017, Art. no. 012161.

[31] A. M. Andrew, ''Multiple view geometry in computer vision,'' *Kybernetes*, vol. 30, pp. 1333–1341, Dec. 2001.

[32] H. Liu, R. Wang, Y. Xia, and X. Zhang, ''Improved cost computation and adaptive shape guided filter for local stereo matching of low texture stereo images,'' *Appl. Sci.*, vol. 10, no. 5, p. 1869, Mar. 2020.

[33] H. Liu, R. Wang, Y. Xia, and X. Zhang, ''Texture category-based matching cost and adaptive support window for local stereo matching,'' *J. Electron. Imag.*, vol. 29, no. 2, Apr. 2020, Art. no. 023026.

[34] M. Menze and A. Geiger, ''Object scene flow for autonomous vehicles,'' in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2015, pp. 3061–3070.

[35] D. Scharstein, H. Hirschmüller, and Y. Kitajima, ''High-resolution stereo datasets with subpixel-accurate ground truth,'' in *Proc. German Conf. Pattern Recognit.*, Sep. 2014, pp. 31–42.

[36] S. Solak and E. D. Bolat, ''A new hybrid stereovision-based distance-estimation approach for mobile robot platforms,'' *Comput. Electr. Eng.*, vol. 67, pp. 672–689, Apr. 2018.

[37] K.-J. Yoon and I. So Kweon, ''Adaptive support-weight approach for correspondence search,'' *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 4, pp. 650–656, Apr. 2006.

[38] D. T. Vu, B. Chidester, H. Yang, M. N. Do, and J. Lu, ''Efficient hybrid tree-based stereo matching with applications to postcapture image refocusing,'' *IEEE Trans. Image Process.*, vol. 23, no. 8, pp. 3428–3442, Aug. 2014.

[39] Y. Jin, H. Zhao, F. Gu, P. Bu, and M. Na, ''A spatial minimum spanning tree filter,'' *Meas. Sci. Technol.*, vol. 32, no. 1, Oct. 2020, Art. no. 015204.

[40] Y. Chai and F. Yang, ''Semi-global stereo matching algorithm based on minimum spanning tree,'' in *Proc. 2nd IEEE Adv. Inf. Manag., Communicates, Electron. Autom. Control Conf. (IMCEC)*, May 2018, pp. 2181–2185.

LONGHAO SUN received the B.S. degree in information and computing science from the Qilu University of Technology, Jinan, China, in 2022. He is currently pursuing the M.S. degree with Harbin Engineering University, Harbin, China. His current research interests include stereo matching and computer vision.